Máquina de Turing, problema da parada e incompletude em sistemas formais

07 – Breve nota sobre a tese de Church

Olá a todos, meu nome é Vinicius. Neste episódio eu falarei brevemente sobre a chamada ***Tese de Church***.

Até o momento, nessa série, nós vimos o conceito de máquina de Turing (MT) – fizemos uma possível formalização desse conceito, que nos dá um entendimento mais preciso do que seria um procedimento efetivo – uma sequência de instruções que, se seguida passo a passo, sem criatividade, consegue-se resolver determinado problema. Vimos ainda o que significa dizer que uma função é *turing-computável* – estabelecemos algumas convenções sobre máquinas de Turing e especificamos como uma MT calcula ou computa uma função.

Vocês podem ter percebido (e eu mesmo acabei percebendo isso depois) que em alguns momentos eu falava sobre a "turing-computabilidade" de funções e em outros eu falava simplesmente de "computabilidade". Eu acabei utilizando esses dois termos como sinônimos em virtude do fato de que eu já tinha em mente a veracidade da chamada *tese de Church* ou ainda *tese de Church-Turing*.

Muito bem, então o que é a tese de Church?

Como eu comentei no primeiro episódio desta série, no início do século XX, o matemático alemão David Hilbert havia proposto um problema relacionado à decidibilidade da lógica de primeira ordem. Este problema nos questiona se existe algum procedimento efetivo que permita decidir se uma dada fórmula da lógica de primeira ordem possui ou não demonstração.

Daí o matemático inglês Alan Turing, resolvendo trabalhar neste problema, parte inicialmente, então, para a tarefa de especificar formalmente o que significa esse conceito de "procedimento efetivo". O resultado dessa sua investigação inicial é o que hoje conhecemos por "máquina de Turing".

Mas Turing não foi o único a se dedicar ao problema da decisão de Hilbert. Outro matemático, o americano Alonzo Church, por exemplo, também estava trabalhando nesse mesmo problema.

Church irá propor uma formalização para a noção de procedimento efetivo que difere daquela proposta pelo Turing, chamada de λ-cálculo. No ano de 1936 ele publica um trabalho provando que vale uma resposta negativa ao problema da decisão de Hilbert.

Só que observe o seguinte: afirmar que vale uma resposta negativa ao problema da decisão é dizer que não existe um procedimento efetivo que resolva esse problema. Mas qual a certeza de que temos que o λ-cálculo de Church de fato abarca todos os procedimentos efetivos possíveis? Será que, de repente, mediante uma outra formalização – a de Turing, por exemplo – não poderíamos responder que na verdade existe um procedimento efetivo para resolver o problema da decisão?

Aqui entra, então a tese de Church, que estabelece a correspondência entre a ideia intuitiva de procedimento efetivo que eu mencionei lá no início – ou seja, um conjunto de instruções para se resolver determinado problema que eu vou seguindo passo a passo – e a noção formal dada pelo λ-cálculo de Church.

Igualando então essas duas noções de computabilidade – uma noção intuitiva e outra formal, isto é, aceitando a tese de church como verdadeira – nós podemos afirmar que o resultado encontrado por Church possui validade universal, digamos assim; não é somente algo que ocorre dentro do formalismo proposto pelo Church.

Mas algo bastante interessante ainda estava por acontecer.

Antes de Church já havia uma noção formal da ideia de computabilidade, dada pelo lógico Kurt Gödel, chamada de *funções recursivas*. Dois alunos do Church conseguiram, no ano de 1935, demonstrar a equivalência das duas formalizações – a de Gödel (funções recursivas) e a de Church (λ-cálculo), o que acrescentou uma boa força à tese de Church (lembrando que o trabalho de Church que eu citei sobre o problema da decisão sairia em 1936).

Nesse mesmo ano de 1936/1937, Turing também sairia o trabalho de Turing sobre o problema da decisão – da mesma forma que Church e de forma independente, Turing demonstrou que o problema da decisão admite uma resposta negativa em relação à sua própria formalização (dada pelas máquinas de Turing, que Turing chamada de "máquinas lógicas"). Além disso, também demonstrou que as noções de máquina de Turing e λ-cálculo são equivalentes – tudo o que pode ser computado por máquina de Turing pode ser computado ou definível no λ-cálculo e vice-versa, ou seja, as duas noções computam ou calculam a mesma classe de funções.

O grande diferencial das máquinas de Turing é que Turing conseguiu quebrar minuciosamente, digamos assim, o processo de computação em 3 componentes fundamentais:

- Ler e escrever símbolos
- Mudança de foco ou de atenção
- Mudança de estado mental

E isso é basicamente o que uma MT faz, conforme a gente pôde ver nas primeiras apresentações da série – ela possui uma fita dividida em quadrados, um cabeçote de escrita e leitura de símbolos, que poderá ler um símbolo por vez que está na fita e escrever nessa fita também; esse dispositivo de leitura/escrita pode se movimentar para a esquerda ou para a direita (mudança de foco) e, por fim, a máquina possui um registrador de estados, que sinaliza um estado atual da máquina.

Eu posso também enunciar a tese de Church-Turing: uma função é computável se e somente se é computável por MT (o que seria equivalente a dizer que uma função é computável se é definível no λ-cálculo, já que vimos que as duas noções são equivalentes). Considerando essa tese verdadeira, eu posso então, em vez de dizer que uma função é "turing-computável", dizer simplesmente que ela é "computável".

Agora, quando Turing faz essa quebra minuciosa do processo de computação naqueles três componentes fundamentais (relembrando: 1 - ler e escrever símbolos, 2 - mudança de foco ou de atenção e 3 - mudança de estado mental), podemos talvez perceber com uma maio clareza o que está em jogo na tese de church-turing – que é o fato de uma função matemática poder ser calculada por meio de um processo tão simples, tão fundamental, de leitura e escrita de símbolos.

Para finalizar, vale citar também que, depois de Gödel, Church e Turing, várias outras formalizações da noção de computabilidade foram propostas, por exemplo, os sistemas de manipulação de símbolos de Post, a computabilidade de Markov, as máquinas de computação URM, propostas por Shepherdson e Sturgis, e assim por diante. Todas essas formalizações calculam a mesma classe de funções.

Como eu disse lá no início, esse é o 7º episódio da série "máquinas de turing, problema da parada e incompletude em sistemas forma", ainda teremos mais 5 episódios, nos quais eu apresentarei uma prova da indecidibilidade do problema da parada, mostrarei como esse problema está relacionado com a incompletude em sistemas formais e ainda mostarrei também como o 10º problema de Hilbert (problema das equações diofantinas) pôde ser resolvido por meio de uma redução ao problema da parada.

Sugestões de leitura:

Artigos

- Aspectos da tese de Church-Turing. Jacob Zimbarg Sobrinho http://rmu.sbm.org.br/Conteudo/n06/n06_Artigo01.pdf
- A tese de Church-Turing. Bruno Loff. http://revistas.rcaap.pt/boletimspm/article/view/3870/2910

Livros:

- Computabilidade, funções computáveis, lógica e os fundamentos da matemática. Walter Carnielli e Richard Epstein.
- Computability, an introduction to recursive function theory. N. J. Cutland.

Obrigado por ouvir o podcast, e até o próximo episódio.

****

NÚMERO IMAGINÁRIO

numeroimaginario.com.br

Contato – vinicius@numeroimaginario.com.br

Padrim – padrim.com.br/numeroimaginario

Podcast – numeroimaginario.com.br/categoria/podcast

Moodle – numeroimaginario.com.br/moodle

Youtube – youtube.numeroimaginario.com.br

Redes Sociais

facebook.com/podcastnumeroimaginario

twitter.com/num_imaginario

****